Wireshark

How to sniff packets very goodly

High Level Overview

- Networked communications involve sending data over the wire
- It can be useful and/or entertaining to see which exact bits get sent
  - Maybe you want to reverse engineer a protocol
  - Maybe you want to watch innocent people send their passwords
  - Maybe you want to perform analytics
- Wireshark is a utility that monitors and inspects networking packets
  - All packets can be dumped, so you can see what bits are in which packets
  - You can also see which applications are sending the data

Example #1: Basic UDP Client & Server

- In which a basic socket sends a paragraph of text to a server, which sends a different response back
- We will use this to understand what a packet is, what a packet header is, and the basics of the Wireshark UI

Example #1: Client Source

```
import socket as z
import sys
s = None
for r in z.getaddrinfo(sys.argv[1], int(sys.argv[2]), z.AF_UNSPEC, z.SOCK_DGRAM):
     af, socktype, proto, canonname, sa = r
     try:
          s = z.socket(af, socktype, proto)
     except OSError as msg:
          s = None; continue
     try:
          s.connect(sa)
     except OSError as msg:
          s.close(); s = None; continue
     break
if s is None:
     print('could not open socket'); sys.exit(1)
s.sendall(b'Hello, world'); data = s.recv(1024); s.close()
print('Received', repr(data))
```

Example #1: Server Source

```
import socket as z
import sys
s = None
for r in z.getaddrinfo(None,int(sys.argv[1]),z.AF_UNSPEC,z.SOCK_DGRAM,0,z.AI_PASSIVE):
     af, socktype, proto, canonname, sa = r
     try:
          s = z.socket(af, socktype, proto)
     except OSError as msg:
          s = None; continue
     try:
          s.bind(sa)
     except OSError as msg:
          s.close(); s = None; continue
     break
if s is None:
     print('Could Not Open Socket'); sys.exit(1)
while True:
     (data, addr) = s.recvfrom(1024)
     if not data: break
     s.sendto(data, addr)
```

Example #1: Demo

OSI Model

- Networking protocols are like onions: they are layered

	Data Unit	Layer	Function	Example
Host Layers	Data	Application	Network process to application	NNTP, SMTP, HTTP, FTP, Gopher, DHCP, DNS
		Presentation	Data representation and encryption	MIME, XDR
		Session	Interhost communication	SPDY, SOCKS, SAP, PPTP, RTP, NetBIOS
	Segments	Transport	End-to-end connections, reliability, flow control	UDP, TCP, DCCP, SCTP
Media Layers	Packet	Network	Path determination and logical addressing	IPv4, IPv6, ICMP, IPsec
	Frame	Data Link	Physical addressing	PPP, IEEE 802.2, MAC
	Bit	Physical	Media and signals	IEEE 802.11, IEEE 802.3

Packet Headers: IPv4

	0		1		2		3
	4b	**4b**	**6b**	**2b**	**3b**	**5b**	**8b**
0	Version	IHL	DSCP	ECN	Length		
4	Identification				Flags	Frag Offset	
8	Time to Live		Protocol		Checksum		
12	Source Address						
16	Destination Address						
20	Options (optional if IHL > 5)						
24							
28							
32							

Packet Headers: UDP vs TCP

- If you were paying attention, there was some extra text being sent/received
- They contain additional information to help the packet move around
- UDP packets contain a size and checksum for data integrity
- TCP packets contain a lot of other data regarding the TCP protocol

Octet	0	1	2	3
0	Src Port		Dest Port	
4	Size		Checksum	
8	Data			
12...				

Octet	0	1	2	3
0	Src Port		Dest Port	
4	Syn			
8	Ack			
12	Offset/Flags		Win Size	
16	Checksum		Urgent?	
20	Options, Padding, Data			
24...				

Example #1: Packet Headers

- Given the fact that E1 used UDP, we can fill out a sample header used by the code and compare it against what Wireshark says
- We know the port of the server, the content of the data, its size, and we can calculate the checksum

Byte	0	1	2	3
0	Determined at Runtime		9001	
4	243		Calculated at Runtime	
8	"According to all known laws of aviation, there is no way a bee should be able to fly. Its wings are too small to get its fat little body off the ground. The bee, of course, flies anyway, because bees don't care what humans think is impossible."			
12...				

https://github.com/DragonFlyBSD/DragonFlyBSD/blob/master/sbin/dhclient/packet.c#L55

Example #2: Basic TCP Client & Server

- TCP is a streaming protocol, in contrast to UDP
- We will run a basic TCP client and server
- And then we will take a look at what it looks like in Wireshark

Example #2: Client Source

```
import socket as z
import sys
s = None
for r in z.getaddrinfo(sys.argv[1],int(sys.argv[2]),z.AF_UNSPEC,z.SOCK_STREAM):
     af, socktype, proto, canonname, sa = r
     try:
          s = z.socket(af, socktype, proto)
     except OSError as msg:
          s = None; continue
     try:
          s.connect(sa)
     except OSError as msg:
          s.close(); s = None; continue
     break
if s is None:
     print('could not open socket'); sys.exit(1)
s.sendall(b'Hello, world'); data = s.recv(1024); s.close()
print('Received', repr(data))
```

Example #2: Server Source

```
import socket as z
import sys
s = None
for r in z.getaddrinfo(None,int(sys.argv[1]),z.AF_UNSPEC,z.SOCK_STREAM,0,z.AI_PASSIVE):
    af, socktype, proto, canonname, sa = r
    try:
      s = z.socket(af, socktype, proto)
    except OSError as msg:
      s = None; continue
    try:
      s.bind(sa); s.listen(1)
    except OSError as msg:
      s.close(); s = None; continue
    break
if s is None:
    print('Could Not Open Socket'); sys.exit(1)
conn, addr = s.accept(); print('Connected by', addr)
while True:
    data = conn.recv(1024)
    if not data: break
    conn.send(data)
conn.close()
```

Example #2: Demo

Example #2: Packet Headers

- As an exercise, use Wireshark to determine what the various components are of the TCP packets are

Octet	0	1	2	3
0	?		9001	
4	Syn			
8	Ack			
12	Offset/Flags		Win Size	
16	Checksum		Urgent?	
20	Options, Padding, Data			
24...				

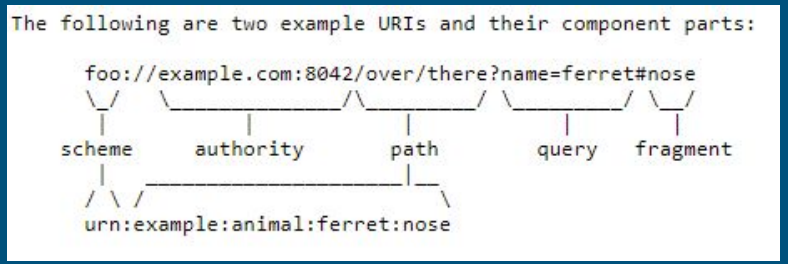
Example #3: Basic HTTP Requests

- HTTP is an application layer protocol
- Sits on top of TCP
- Generally runs on port 80
- Let's screw around with it

Example #3: Basic HTTP Requests

Client sends:

```
GET / HTTP/1.1

Host: bing.com
```

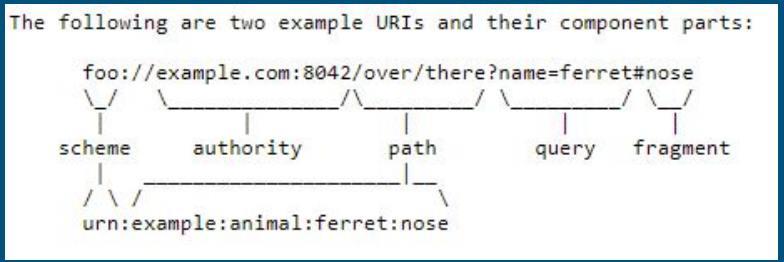


Notice that the request ends in a “\r\n\r\n”

Example #3: Basic HTTP Requests

Server responds:

```
HTTP/1.1 200 OK

Content-Type: text/html

Content-Length: 47


<html>

     <body>Hello World!</body>

</html>
```



Example #3: Client Source

```
import requests as r
import sys

print(r.get(sys.argv[1]).text)

# e.g. http://openbsd.org
```

Example #3: Demo

Example #4: Basic HTTPS Requests

- Another similar application level protocol
- Combines HTTP with TLS to encrypt data in transit
- Rerun the past demo, but go to a website that uses HTTPS instead of HTTP

Example #4: Demo